Abstract

We report on the *Equational Theories Project* (ETP), an online collaborative pilot project to explore new ways to collaborate in mathematics with machine assistance. The project sought to determine the implication graph between 4692 equational laws on magmas, by a combination of human-generated and automated proofs, all validated by the formal proof assistant language *Lean*. **state key outcomes**

The Equational Theories Project

Matthew Bolan, Jose Brox, Mario Carneiro, Andrés Goens, Zoltan Kocsis, Alex Meibu

November 23, 2024

Contents

1 Introduction

The purpose of this paper is to report on the *Equational Theories Project* (ETP)[1], a pilot project launched[2] in September 2024 to explore new ways to collaboratively work on mathematical research projects using machine assistance. The project goal, in the area of universal algebra, was selected to be particularly amenable to crowdsourced and computer-assisted techniques, while still being of mathematical research interest. **Describe outcomes**

1.1 Magmas and Equational Laws

In order to describe the mathematical goals of the ETP, we need some notation. A *magma* $M = (M, \diamond)$ is a set M (known as the *carrier*) together with a binary operation $\diamond \colon M \times M \to M$. An *equational law* for a magma, or *law* for short, is an identity involving $\diamond$ and some indeterminates, which we will typically denote using the symbols x, y, z, u, v, w. In the ETP, a unique number was assigned to each equational law, via a numbering system that we describe in Appendix A. For instance, the *commutative law* $x \diamond y = y \diamond x$ is assigned the equation number (E43), while the *associative law* $(x \diamond y) \diamond z = x \diamond (y \diamond z)$ is assigned the

[1]`https://teorth.github.io/equational_theories/`
[2]`https://terrytao.wordpress.com/2024/09/25`

equation number (E4512). A list of all equations referred to by number in this paper is provided in Appendix A.

Formally, one can represent an equational law syntactically as a string $w_1 \simeq w_2$, where w_1, w_2 are words in a free magma generated by formal indeterminate symbols; see Section 2.

A magma M obeys a law E if the law E holds for all possible assignments of the indeterminate to M, in which case we write $M \models E$. Thus for instance $M \models E43$ if one has $x \diamond y = y \diamond x$ for all $x, y \in M$.

We say that a law E *entails* or *implies* another law E' if every magma that obeys E, also implies E': $(M \models E) \implies (M \models E')$. We write this relation as $E \leq E'$ or $E \vdash E'$; it is easily seen to be a pre-order. We say that two laws are *equivalent* if they entail each other. For instance, the constant law $x \diamond y = z \diamond w$ (E46) can easily be seen to be equivalent to the law $x \diamond x = y \diamond z$ (E41).

In this pre-ordering, the maximal element is given by the trivial law $x = x$ (E1), and the minimal element is given by the singleton law $x = y$ (E2).

The *order* of an equational law is the number of occurrences of the magma operation. For instance, the commutative law (E43) has order 2, while the associative law (E4512) has order 4. We note some selected laws of small order that have previously appeared in the literature:

- The *central groupoid law* $x = (y \diamond x) \diamond (x \diamond z)$ (E168) is an order 3 law introduced by Evans [4] and studied further by Knuth [11] and many further authors, being closely related to central digraphs (also known as unique path property diagraphs), and leading in particular to the discovery of the Knuth-Bendix algorithm [12]; see [14] for a more recent survey.
- *Tarski's axiom* $x = y \diamond ((z \diamond (x \diamond (y \diamond z))))$ (E543) is an order 4 law that was shown by Tarski [24] to characterize the operation of subtraction in an abelian group; that is to say, a magma M obeys (E543) if and only if there is an abelian group structure on M for which $x \diamond y = x - y$ for all $x, y \in M$.
- In a similar vein, it was shown in [20] (see also [21]) that the order 4 law $x = (y \diamond z) \diamond (y \diamond (x \diamond z))$ (E1571) characterizes addition (or subtraction) in an abelian group of exponent 2; it was shown in [18] that the order 4 law $x = (y \diamond ((x \diamond y) \diamond y)) \diamond (x \diamond (z \diamond y))$ (E345169) characterizes the Sheffer stroke in a boolean algebra, and it was shown in [7] that the order 8 law $x = y \diamond ((((y \diamond y) \diamond x) \diamond z) \diamond (((y \diamond y) \diamond y) \diamond z))$ (E42323216) characterizes division in a (not necessarily abelian) group.

Some further examples of laws characterizing well-known algebraic structures are listed in [17].

The Birkhoff completeness theorem [2, Th. 3.5.14] implies that an implication $E \vdash E'$ of equational laws holds if and only if the left-hand side of E' can be transformed into the right-hand side by a finite number of substitution rewrites using the law E. However, the problem of determining whether such an implication holds is undecidable in general [19]. Even when the order is small, some implications[3] can require lengthy computer-assisted proofs; for instance, it was noted in [9] that the order 4 law $x = (y \diamond x) \diamond ((x \diamond z) \diamond z)$ (E1689) was equivalent to the singleton law (E2), but all known proofs are computer-assisted.

1.2 The Equational Theories Project

As noted in Appendix A, there are 4694 equational laws of order at most 4. The primary mathematical goal of the ETP was to completely determine the implication pre-ordering $\leq$ for this set of laws. Such systematic determinations of implication graphs have been seen previously in the literature; for instance, in [22], the relations between 60 identities of Bol–Moufang type were established, and in the blog post [25, §17], some initial steps towards generating this graph for the first hundred or so laws on our list were performed. However, to our knowledge, the ETP is the first project to study such implications at the scale of thousands of laws.

The ETP requires the determination of the truth or falsity of $4694^2 = 22033636$ implications; while one can use properties such as the transitivity of the pre-ordering to reduce the work somewhat, this is clearly a task that requires significant automation. It was also a project highly amenable to crowdsourcing, in which different participants could work on developing different techniques, each of which could be used to fill out a different part of the implication graph. In this respect, the project could be compared with a Polymath project [6], which used online forums such as blogs and wikis to openly collaborate on a mathematical research problem. However, the Polymath model required human moderators to review and integrate the contributions of the participants, which clearly would not scale to the ETP which required the verification of over twenty million mathematical statements. Instead, the ETP was centered around a Github repository in which the formal mathematical contributions had to be entered in the proof assistant language *Lean*, where they could be automatically verified. In this respect, the ETP was more similar to the recently concluded Busy Beaver Challenge[4], which was a similarly crowdsourced project that computed the fifth Busy Beaver number $BB(5)$ to be 47176870 through an analysis of 88664064 Turing machines, with the halting analysis being verified in a variety of computer languages, with the final formal proof written in the proof assistant language *Coq*. One of the aims of the ETP was to explore potential workflows for such collaborative, formally verified mathematical research projects that could serve as a model for future projects of this nature.

Secondary aims of the ETP included the possibility of discovering unusually interesting equational laws, or new experimental observations about such laws, that had not previously been noticed in the literature; and to develop benchmarks to assess the performance of automated theorem provers and other AI tools.

[3]Another contemporaneous example of this phenomenon was the solution of the Robbins problem [16].

[4]`https://bbchallenge.org/`

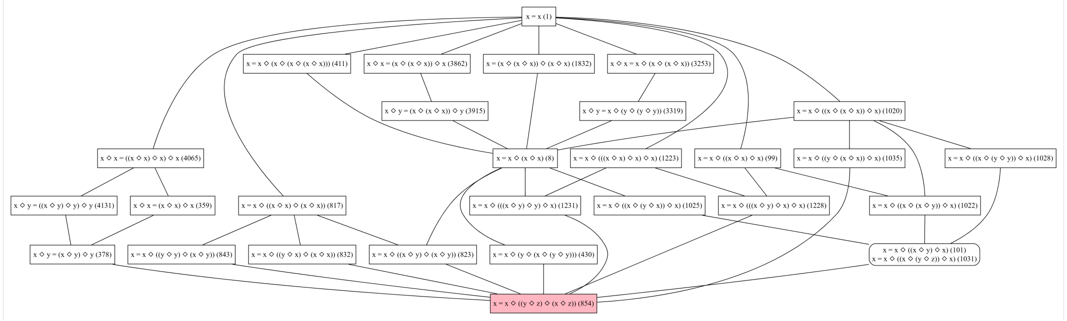
x = x (1)
x = x ◇ (x ◇ (x ◇ (x ◇ x))) (411)
x ◇ x = (x ◇ (x ◇ x)) ◇ x (3862)
x = (x ◇ (x ◇ x)) ◇ (x ◇ x) (1832)
x ◇ x = x ◇ (x ◇ (x ◇ x)) (3253)
x ◇ y = (x ◇ (x ◇ x)) ◇ y (3915)
x ◇ y = x ◇ (y ◇ (y ◇ y)) (3319)
x = x ◇ ((x ◇ (x ◇ x)) ◇ x) (1020)
x ◇ x = ((x ◇ x) ◇ x) ◇ x (4065)
x = x ◇ (x ◇ x) (8)
x = x ◇ (((x ◇ x) ◇ x) ◇ x) (1223)
x = x ◇ ((x ◇ x) ◇ x) (99)
x = x ◇ ((y ◇ (x ◇ x)) ◇ x) (1035)
x = x ◇ ((x ◇ (y ◇ y)) ◇ x) (1028)
x ◇ y = ((x ◇ y) ◇ y) ◇ y (4131)
x ◇ x = (x ◇ x) ◇ x (359)
x = x ◇ ((x ◇ x) ◇ (x ◇ x)) (817)
x = x ◇ (((x ◇ y) ◇ y) ◇ x) (1231)
x = x ◇ ((x ◇ (y ◇ x)) ◇ x) (1025)
x = x ◇ (((x ◇ y) ◇ x) ◇ x) (1228)
x = x ◇ ((x ◇ (x ◇ y)) ◇ x) (1022)
x ◇ y = (x ◇ y) ◇ y (378)
x = x ◇ ((y ◇ y) ◇ (x ◇ y)) (843)
x = x ◇ ((y ◇ x) ◇ (x ◇ x)) (832)
x = x ◇ ((x ◇ y) ◇ (x ◇ y)) (823)
x = x ◇ (y ◇ (x ◇ (y ◇ y))) (430)
x = x ◇ ((x ◇ y) ◇ x) (101)
x = x ◇ ((x ◇ (y ◇ z)) ◇ x) (1031)
x = x ◇ ((y ◇ z) ◇ (x ◇ z)) (854)

Figure 1: A Hasse diagram of all the equational laws implied by (E854). An edge in this diagram indicates that the lower equation implies the higher one. Rounded rectangles indicate groups of equivalent laws. This graph was produced by the visualization tool *Graphiti*, which was developed for this project.

1.3 Outcomes

The ETP achieved its primary objective, with all of the implications formalized in the proof assistant language *Lean*, and can be found on the ETP GitHub repository. See Figure 1 for a small fragment of the implication graph produced. The experience of running such a large collaborative research project introduced several challenges, which we report upon in Section 4. Also, a variety of methods with varying degrees of automation or computer-assistance had to be developed to resolve all the implications, which had quite a variety of difficulty levels.

Of the 22033636 possible implications $E \vdash E'$, 8178279 (or 37.12%) would end up being true. To establish such positive implications $E \vdash E'$, the main techniques used were as follows:

- A very small number of positive implications were established and formalized by hand, mostly through direct rewriting of the laws; but this approach would not scale to the full project.
- Simple rewriting rules, for instance based on the observation that any law of the form $x = f(y, z, \dots)$ was necessarily equivalent to the trivial law (E2), could already reduce the size of potential equivalence classes by a significant fraction. We discuss this method in Section 6.1.
- The preorder axioms for $\vdash$, as well as the "duality" symmetry of the preorder with respect to replacing a magma operation $x \diamond y$ with its opposite $x \diamond^{\mathrm{op}} y := y \diamond x$, can be used to significantly cut down on the number of implications that need to be proven explicitly; ultimately, only 10657 (0.05%) of the positive implications needed a direct proof.
- Automated Theorem Provers (ATP) could be deployed at acceptable compute cost to establish this generating set of positive implications, as discussed in Section 7.

More challenging were the 13855357 (62.88%) implications that were false, $E \not\vdash E'$. Here, the range of techniques needed to refute such implications were quite varied.

- Syntactic methods, such as observing an "matching invariant" of the law E that was not shared by the law E', could be used to obtain some refutations. For instance, if both sides of E had the same order, but both sides of E' did not, this could be used to syntactically refute $E \vdash E'$. Similarly, if the law E was confluent, enjoyed a complete rewriting system, or otherwise permitted some understanding of the free magma associated to that law, one could decide the assertions $E \vdash E'$ for all possible laws E', or at least a significant fraction of such laws. We discuss these methods, and the extent to which they can be automated in Section 6. One novel such invariant we introduce here is the "twisting semigroup" of an equational theory, which gave simple refutations of some otherwise very challenging implications.

- Small finite magmas, which can be described explicitly by multiplication tables, could be tested by brute force computations to provide a large number of counterexamples to implications, or by ATP-assisted methods. See Section 5.1.

- Linear models, in which the magma operation took the form $x \diamond y = ax + by$ for some (commuting or non-commuting) coefficients a, b, allowed for another large class of counterexamples to implications, which could be automatically scanned for either by brute force or by Grobner basis type calculations. See Section 5.2.

- Translation invariant models, in which the magma operation took the form $x \diamond y = x + f(y - x)$ on an additive group, or $x \diamond y = xf(x^{-1}y)$ on a non-commutative group, reduce matters to analyzing certain functional equations; see Section 5.3.

- Greedy methods, in which either the multiplication table $(x, y) \mapsto x \diamond y$ or the function f determining a translation-invariant model are iteratively constructed by a greedy algorithm subject to a well-chosen ruleset, were effective in resolving many implications not easily disposed of by preceding methods. See Section 5.5.

- Starting with a simple base magma M obeying both E and E', and either enlarging it to a larger magma $M' \supset M$, extending it to a magma N with a projection homomorphism $\pi : N \to M$, or modifying the multiplication table on a small number of values, also proved effective when combined with greedy methods. See Section 5.6.

- Some *ad hoc* models based on existing mathematical objects, such as infinite trees, rings of polynomials, or "Kisielewicz models" utilizing the prime factorization of the natural numbers, could also handle some otherwise difficult cases. In some cases, the magma law induced some relevant and familiar structures, such as a directed graph or a partial order, which also helped guide counterexample constructions. See Section 5.4.

- Automated theorem provers were helpful in identifying which simplifying axioms could be added to the magma without jeopardizing the ability to refute the desired implication $E \vdash E'$.

1.4 Extensions

While the primary objective of the ETP was being completed, some additional related results were generated as spinoffs. Specifically:

- In Section 5.1 we report on a variant of the implication graph of the original set of 4692 equations, in which the magma is required to be finite.
- In Section 9 we report on classifying which of the 57882 distinct laws of order 5 are equivalent to the singleton law (E2), either with or without the requirement that the magma be finite.
- In Section 10 we report on classifying the laws of order 8 that are equivalent to the Higman-Neumann law (E42323216).

Also mention ML stuff, GUI

2 Notation and Mathematical Foundations

If M is a magma, we define the left and right multiplication operators $L_a, R_a \colon M \to M$ for $a \in M$ by the formula

$$L_x y = R_y x := x \diamond y. \tag{1}$$

We also define the squaring operator $S \colon M \to M$ by

$$Sx := x \diamond x = L_x x = R_x x \tag{2}$$

and the (right) cubing operator $C \colon M \to M$ by

$$Cx := Sx \diamond x = R_x^2 x. \tag{3}$$

If X is an alphabet, we let M_X denote the free magma generated by X, thus an element of M_X is either a letter in X, or of the form $w_1 \diamond w_2$ with $w_1, w_2 \in M_X$. Every function $f \colon X \to M$ into a magma M extends to a unique homomorphism $\varphi_f \colon M_X \to M$, that is to say a function for which $\varphi_f(w \diamond w') = \varphi_f(w) \diamond \varphi_f(w')$ for all $w, w' \in M_X$. Formally, an equational law with some indeterminates in X can be written as $w_1 \simeq w_2$ for some $w_1, w_2 \in X$; a magma M then obeys this law if and only if $\varphi_f(w_1) = \varphi_f(w_2)$ for all $f \colon X \to M$. We also define the order of a word $w \in M_X$ to be the number of occurrences of $\diamond$ in the word, thus letters in X are of order 0, and the order of $w_1 \diamond w_2$ is the sum of the orders of w_1, w_2.

A *theory* is a collection Γ of equational laws; we say that a magma M *satisfies* a theory, and write $M \models \Gamma$, if every law in Γ is obeyed by M. If E is an equational law, we write $\Gamma \vdash E$ if every magma that satisfies Γ also satisfies E. A *free magma* $M_{X,\Gamma}$ for such a theory Γ and an alphabet X is a magma satisfying Γ together with a map $\iota_{X,\Gamma}\colon X \to M_{X,\Gamma}$ which is universal in the sense that every function $f\colon X \to M$ to a magma M satisfying Γ uniquely determines a homomorphism $\varphi_{f,\Gamma}\colon M_{X,\Gamma} \to M$ such that $\phi_{f,\Gamma} \circ \iota_{X,\Gamma} = f$. This magma is unique up to isomorphism; a canonical way to construct it is to quotient M_X by the equivalence relation given by declaring w, w' to be equivalent if $\Gamma \vdash w \simeq w'$. If $\Gamma = \{E\}$ consists of a single law E, we write $M_{X,E}$ for $M_{X,\{E\}}$ and $\phi_{f,E}$ for $\phi_{f,\{E\}}$. **Give reference for free magmas relative to theories**

In general, the free magma $M_{X,\Gamma}$ is difficult to describe in a tractable form, but for some theories, one has a simple description:

Example 2.1 (Commutative and associative free magma)**.** The free magma $M_{X,\{E43,E4512\}}$ for the commutative law (E43) and the associative law (E4512) is the free abelian semigroup generated by X (with $\iota_{X,\{E43,E4512\}}$ the obvious embedding map).

Example 2.2 (Left-absorptive free magma)**.** The free magma $M_{X,\{E4\}}$ for the left-absorptive law (E4) is the magma with carrier X and operation $x \diamond y = x$ (with ι_{X,E_4} the identity).

Every magma M has an opposite M^{op}, which has the same carrier but the opposite operation $x \diamond^{\mathrm{op}} y := y \diamond x$. A magma M obeys an equational law E if and only if its opposite M^{op} obeys the dual law E^*, defined by reversing the all operations. For instance, the dual of $x \diamond y = x \diamond (y \diamond z)$ (E327) is $y \diamond x = (z \diamond y) \diamond x$, which in our numbering system we rewrite in normal form as $x \diamond y = (z \diamond x) \diamond y$ (E395).

We then see that the implication graph has a duality symmetry: given two equational laws E_1, E_2, we have $E_1 \vdash E_2$ if and only if $E_1^* \vdash E_2^*$.

3 Formal Foundations

TODO: expand this sketch.

Here we describe the Lean framework used to formalize the project, covering technical issues such as:

- Magma operation symbol issues
- Syntax (‘LawX‘) versus semantics (‘EquationX‘)
- ”Universe hell” issues
- Additional verification (axiom checking, Leanchecker, etc.)
- Use of the ‘conjecture‘ keyword

- Use of namespaces to avoid collisions between contributions. (Note: we messed up with this with FreeMagma! Should have namespaced back end results as well as front end ones.)
- Use of Facts command to efficiently handle large numbers of anti-implications at once

Upstream contributions:

- Mathlib contributions
- LeanBlueprint contributions

4 Project Management

TODO: expand this sketch.

Shreyas Srinivas and Pietro Monticone have volunteered to take the lead on this section.

Discuss topics such as:

- Project generation from template
- Github issue management with labels and task management dashboard
- Continuous integration (builds, blueprint compilation, task status transition)
- Pre-push git hooks
- Use of blueprint (small note, see #406: blueprint chapters should be given names for stable URLs)
- Use of Lean Zulip (e.g. use of polls)

Maybe give some usage statistics, e.g. drawing from `https://github.com/teorth/equational_theories/actions/metrics/usage`

Mention that FLT is also using a similar workflow.

4.1 Handling Scaling Issues

Also mention some early human-managed efforts (”outstanding tasks”, manually generated Hasse diagram, etc.) which suffices for the first one or two days of the project but rapidly became unable to handle the scale of the project.

Mention that some forethought in setting up a Github organizational structure with explicit admin roles etc. may have had some advantages if we had done so in the planning stages of the project, but it was workable without this structure (the main issue is that a single person - Terry - had to be the one to perform various technical admin actions).

Use of transitive reduction etc. to keep the Lean codebase manageable. Note that the project is large enough that one cannot simply accept arbitrary amounts of Lean code into the codebase, as this could make compilation times explode. Also note somewhere that transitive completion can be viewed as directed graph completion on a doubled graph consisting of laws and their formal negations.

Technical debt issues, e.g., complications stemming from an early decision to make Equations.lean and AllEquations.lean the ground truth of equations for other analysis and visualization tools, leading to the need to refactor when AllEquations.lean had to be split up for performance reasons.

Note that the ”blueprint” that is now standard for guiding proof formalization projects is a bit too slow to keep up with this sort of project that is oriented instead about proving new results. Often new results are stated and formalized without passing through the blueprint, which is then either updated after the fact, or not at all. So the blueprint is more of an optional auxiliary guiding tool than an essential component of the workflow.

4.2 Other Design Considerations

Explain what ”trusting” Lean really means in a large project. Highlight the kind of human issues that can interfere with this and how use of tools like external checkers and PR reviews by people maintaining the projects still matters. Provide guidelines on good practices (such as branch protection or watching out for spurious modifications in PRs, for example to the CI). Highlight the importance of following a proper process for discussing and accepting new tasks, avoiding overlaps etc. These issues are less likely to arise in projects with one clearly defined decision maker as in this case, and more likely to arise when the decision making has to be delegated to many maintainers.

Note that despite the guarantees provided by Lean, non-Lean components still contained bugs. For instance, an off-by-one error in an ATP run created a large number of spurious conjectures, and some early implementations of duality reductions (external to Lean) were similarly buggy. ”Unit tests”, e.g., checking conjectured outputs against Lean-validated outputs, or by theoretical results such as duality symmetry, were helpful, and the Equation

Explorer visualization tool also helped human collaborators detect bugs.

Meta: documenting thoughts for the future record is quite valuable. It would be extremely tedious to try to reconstruct real-time impressions long after the fact just from the github commit history and Zulip chat archive.

4.3 Maintenance

Describe the role of maintainers and explain why they need to be conversant in the mathematics being formalised, as well as Lean. As such, the role of maintainers is often akin to postdocs or assistant profs in a research group who do some research of their own, but spend much of their time to guide others in performing their tasks, the key difference being that contributors are volunteers who choose their own tasks. Explain the tasks maintainers must perform. Examples:

- Reviewing proofs,
- Helping with proofs and theorem statements when people get stuck,
- Offering suggestions and guidance on how to produce shorter or more elegant proofs,
- Ensuring some basic standards are met in proof blocks that make proofs robust to upstream changes,
- Creating and maintaining CI processes,
- Responding to task requests,
- Evaluating theorem and definition formulations (for example unifying many theorem statements into one using FactsSyntax),
- Suggesting better ones where possible,
- Ensuring that there is no excessive and pointless overlap of content in different contributions (TODO: elaborate on what level of overlap was permissible and what we consider excessive).

5 Counterexample constructions

In this section we collect the various techniques developed in the ETP to construct counterexamples to various implications $E \vdash E'$.

5.1 Finite magmas

TODO: Expand this sketch

Discuss semi-automated creation of finite counterexamples (as discussed here) Describe various sources of example magmas used in counterexamples, including the ones listed here.

Also note some "negative results" - classes of finite magmas that did not yield many additional refutations, e.g. commutative 5x5 magmas.

Mention finite immunity

Using SAT solvers to find medium sized finite magmas obeying a given law? See this discussion.

Discuss computational and memory efficiencies needed to brute force over extremely large sets of magmas. SAT solving may be a better approach past a certain size!

5.2 Linear models

A fruitful source of counterexamples is the class of *linear magmas*, where the carrier M is a ring (which may be commutative or non-commutative, finite or infinite), and the operation $\diamond$ is given by $x \diamond y = ax + by$ for some coefficients $a, b \in M$; one can also generalize this slightly to *affine magmas*, in which the operation is given by $x \diamond y = ax + by + c$, but for simplicity we shall focus on linear magmas here. It is easy to see that in a linear magma, any word $w(x_1, \dots, x_n)$ of n indeterminates also takes the linear form

$$w(x_1, \dots, x_n) = \sum_{i=1}^{n} P_{w,i}(a,b) x_i$$

for some (possibly non-commutative) polynomial $P_{w,i}$ in a, b with integer coefficients. Thus, a linear magma will obey an equational law $w_1 \simeq w_2$ if and only if the pair (a, b) lies in the (possibly non-commutative) variety

$$\{(a, b) \in M \times M : P_{w_1,i}(a, b) = P_{w_2,i}(a, b) \text{ for all } i\}. \tag{4}$$

As such, a necessary condition for such a law $w_1 \simeq w_2$ to entail another law $w_1' \simeq w_2'$ is that one has the inclusion

$$\{(a, b) \in M \times M : P_{w_1,i}(a, b) = P_{w_2,i}(a, b) \text{ for all } i\} \subset \{(a, b) \in M \times M : P_{w_1',i}(a, b) = P_{w_2',i}(a, b) \text{ for all } i\}$$

for all rings M. For commutative rings, this criterion can be checked by standard Grobner basis techniques; in the noncommutative case one can use methods such as the diamond lemma [3].

Example 5.1 (Commutative counterexample)**.** For the law $x = y \diamond (((x \diamond y) \diamond x) \diamond y)$ (E1286), the variety (4) can be computed to be

$$\{(a, b) \in M \times M : 1 = a + ba^3 + bab, 0 = a + ba^2b + b^2\}$$

while the variety for the idempotent law (E3) is

$$\{(a, b) \in M : a + b = 1\}.$$

Thus to show that (E1286) does not entail (E3), it suffices to locate elements a, b of a ring M for one has $1 = a + ba^3 + bab$, $0 = a + ba^2b + b^2$, and $a + b \neq 1$. Here one can take a commutative example, for instance when $M = \mathbb{Z}/p\mathbb{Z}$ and $(p, a, b) = (11, 1, 7)$.

Example 5.2 (Noncommutative counterexample)**.** For the law $x = y \diamond ((y \diamond (x \diamond z)) \diamond z)$ (E1117), the variety (4) can be computed to be

$$\{(a, b) \in M \times M : 1 = baba, 0 = a + ba^2, 0 = bab^2 + b^2\}$$

while the variety for $x = (x \diamond ((x \diamond x) \diamond x)) \diamond x$ (E2441) is

$$\{(a, b) \in M \times M : a^2 + aba^2 + abab + ab^2 + b = 1\}.$$

Observe that if $ba = -1$, then (a, b) automatically lies in the first set, and lies in the second set if and only if $(ab+1)(b-1) = 0$. One can then show that (E1117) does not imply (E2441) by setting $a = L$, $b = -R$ where L, R are the left and right shift operators respectively on the ring of integer-valued sequences $\mathbb{Z}^{\mathbb{N}}$. With some *ad hoc* effort one can convert this example into a less linear, but simpler (and easier to formalize) example, namely the magma with carrier $\mathbb{Z}$ and operation $x \diamond y = 2x - \lfloor y/2 \rfloor$.

Remark 5.3. As essentially observed in [1], if there is a commutative linear counterexample to an implication $E \vdash E'$, then by the Lefschetz principle this counterexample can be realized in a finite field $\mathbb{F}_q$ for some prime power q (and by the Chebotarev density theorem one can in fact take q to be a prime, so that the carrier is of the form $\mathbb{Z}/p\mathbb{Z}$ for some prime p). As such, we have found that an effective way to refute implications by the commutative linear magma method is to simply perform a brute force search over linear magmas $x \diamond y = ax + by$ in $\mathbb{Z}/p\mathbb{Z}$ for various triples (p, a, b). **Discuss performance of this method.**

On the other hand, the refutations obtained by non-commutative linear constructions need not have a finite model. For instance, consider the refutation $E1117 \not\vdash E2441$ from Example 5.2. The law (E1117) can be rewritten as $L_y R_z L_y R_z x = x$. This implies that R_z is injective and L_y is surjective for all y, z. For finite magmas M, this then implies that the L_y, R_z are in fact invertible, and hence we have also $R_z L_y R_z L_y x = x$, which implies (E2441) by setting $x = y = z$. Thus the refutation $E1117 \not\vdash E2441$ is "immune" to finite counterexamples.

Remark 5.4. One can also consider nonlinear magma models, such as quadratic models $x \diamond y = ax^2 + bxy + cy^2 + dx + ey + f$ in a cyclic group $\mathbb{Z}/N\mathbb{Z}$. For small values of N, we have found such models somewhat useful in providing additional refutations of implications

$E \vdash E'$ beyond what can be achieved by the linear or affine models. However, as the polynomials associated to a word $w(x_1, \dots, x_n)$ tend to be of high degree (exponential in the order of the word), it becomes quite rare for such models to obey a given equation E when N is large.

Remark 5.5. One can also consider the seemingly more general linear model $x \diamond y = ax + by$, where the carrier M is now an abelian group, and a, b act on M by homomorphisms, that is to say that they are elements of the endomorphism ring $\mathrm{End}(M)$. However, this leads to exactly the same varieties (4) (where M is now replaced by the endomorphism ring $\mathrm{End}(M)$) and so does not increase the power of the linear model for the purposes of refuting implications.

5.3 Translation-invariant models

Translation-invariant magmas (see e.g., this thread for a nice example). Note: any magma with a transitive symmetry will lift to a translation-invariant model, so this helps explain why these are common examples. Also symmetric models could be slightly more likely to obey various laws than general models due to degree of freedom considerations.

Mention translation-invariant immunity

5.4 Ad hoc constructions

Tree based constructions, see here.

5.5 Greedy constructions

5.6 Modifying base models

6 Syntactic arguments

Many proofs or refutations of implications (or equivalences) between two equational laws E, E' can be obtained from the syntactic form of the equation. We discuss some techniques here that were useful in the ETP.

6.1 Simple rewrites

Many equational laws E' can be formally deduced from a given law E by applying the Lean 'rw' tactic to rewrite E' repeatedly by some forward or backward application of E applied to arguments that match some portion of E. For instance, the commutative law (E43) clearly implies $x \diamond (y \diamond z) = (y \diamond z) \diamond x$ (E4531) by a single such rewrite. A brute force application of such rewrite methods is already able to directly generate about $15,000$ such implications, including many equivalences to the singleton law (E2) and the constant law (E46). After applying transitive closure, this generates about four million further such implications.

A simple observation that already generates many equivalences is that any equation of the form $x = f(y, z, \dots)$ necessarily is equivalent to the trivial law $x = y$; similarly, an equation of the form $f(x, y) = g(z, w, \dots)$ implies $f(x, y) = f(x', y')$; and so forth.

6.2 Matching invariants

Fix an alphabet X. An *matching invariant* is an assignment $I \colon M_X \to \mathcal{I}$ of an object $I(w) \in \mathcal{I}$ in some space $\mathcal{I}$ to each word $w \in M_X$ with the property that if an equational law $w_1 \simeq w_2$ has matching invariants $I(w_1) = I(w_2)$, then the same matching $I(w_1') = I(w_2')$ holds for any consequence $w_1' \simeq w_2'$. In particular, if one law $I(w_1) = I(w_2)$ and $I(w_1') \neq I(w_2')$, then the law $w_1 \simeq w_2$ does not imply the law $w_1' \simeq w_2'$.

A simple example of a matching invariant is the multiplicity $(n_x)_{x \in X}$ of variables of a word: if w_1, w_2 have all variables x appear the same number of times n_x in both words, then any rewriting of a word w using the law $w_1 \simeq w_2$ will preserve this property. Hence, if w_1', w_2' do not have the each variable appear the same number of times in both words, then $w_1 \simeq w_2$ cannot imply $w_1' \simeq w_2'$. For instance, the commutative law (E43) cannot imply the left-absorptive law (E4).

One source of matching invariants comes from the free magma M_X of a theory:

Proposition 6.1 (Free magmas and matching invariants)**.** *Let Γ be a theory, and let $\iota_{X,\Gamma} \colon X \to M_{X,\Gamma}$ be the map associated to the free magma $M_{X,\Gamma}$ for that theory. Then the map $I \colon M_X \to M_{X,\Gamma}$ defined by $I(w) := \varphi_{\iota_{X,\Gamma}}(w)$ is an invariant.*

Proof. Suppose that $w_1 \simeq w_2$ entails $w_1' \simeq w_2'$, and that $I(w_1) = I(w_2)$. For any $f \colon X \to M_{X,\Gamma}$, the two maps $\varphi_f, \varphi_{f,\Gamma} \circ \varphi_{\iota_{X,\Gamma}} \colon M_X \to M_{X,\Gamma}$ are both homomorphisms that extend f, hence agree by the universal property of M_X, as displayed by the following commutative diagram:

$$\begin{tikzcd}
& X \arrow[dl, hook'] \arrow[d, "\iota_{X,\Gamma}"'] \arrow[dr, "f"] & \\
M_X \arrow[r, "I=\varphi_{\iota_{X,\Gamma}}"] \arrow[rr, bend right, "\varphi_f"'] & M_{X,\Gamma} \arrow[r, "\varphi_{f,\Gamma}"] & M_{X,\Gamma}
\end{tikzcd}$$

In particular, the hypothesis $I(w_1) = I(w_2)$ implies that $\varphi_f(w_1) = \varphi_f(w_2)$ for all $f \colon X \to M_{X,\Gamma}$; that is to say, the magma $M_{X,\Gamma}$ obeys the law $w_1 \simeq w_2$, and hence also $w_1' \simeq w_2'$ by hypothesis. In particular, $\varphi_{\iota_{X,\Gamma}}(w_1') = \varphi_{\iota_{X,\Gamma}}(w_2')$, which gives $I(w_1') = I(w_2')$ as required. $\square$

Example 6.2. If we take $\Gamma = \{E4\}$ to be the theory of the left-absorptive law (E4) as described in Example 2.2, then the matching invariant $I(w)$ produced by Proposition 6.1 is the left-most letter of the alphabet X appearing in the word; for instance $I((x \diamond y) \diamond z) = x$. Thus, for example, the left-absorptive law (E4) cannot imply the right-absorptive law $x = y \diamond x$ (E5).

Example 6.3. If we take $\Gamma = \{E43, E4512\}$ to be the theory of the commutative law (E43) and the associative law (E4512), then by Example 2.1, the associated invariant $I(w) = \sum_{x \in X} n_x e_x$ is the formal sum of all the generators e_x appearing in the word w, in the free abelian semigroup generated by those generators. This recovers the preceding observation that the multiplicities $(n_x)_{x \in X}$ form a matching invariant.

Example 6.4. Let $n \geq 1$ be a positive integer, and consider the theory $\Gamma = \{E43, E4512, E_n\}$ consisting of the previous theory $\{E43, E4512\}$ together with the order n law $L_x^y x = y$. One can check that the free magma $M_{X,\Gamma}$ can be described as the free group of exponent n with generators $e_x, x \in X$, with associated map $\iota_{X,\Gamma} \colon x \mapsto e_x$. The associated matching invariant $I(w) = \sum_{x \in X} n_x e_x$ is essentially the multiplicities $(n_x \bmod n)_{x \in X}$ modulo n, which gives a slightly stronger criterion than the preceding matching invariant for refuting implications. For example, the cubic idempotent law $x = (x \diamond x) \diamond x$ (E23) has matching invariants $e_x = 3e_x$ in the $n = 2$ case, and hence does not imply the idempotent law $x = x \diamond x$ (E3) since $e_x \neq 2e_x$ in the $n = 2$ case.

Give some statistics on how many refutations can be established by these methods.

6.3 Confluence

Define a confluent law and give some examples.

6.4 Complete rewriting systems

Define a complete rewriting system and give some examples.

6.5 Unique factorization

In general, the free magma $M_{X,E}$ for a given equational law E, which we can canonically define as M_X quotiented by the equivalence relation $E \vdash (w_1' \simeq w_2')$, is hard to descibe explicitly; indeed, from the undecidability of implications, such a magma cannot be computably described for arbitrary E. Nevertheless, for some laws it is possible to obtain some partial understanding of $M_{X,E}$ from a syntactic perspective. For instance, if we can refute the implication $E \vdash (w'_1 \simeq w'_2)$ by other means, then this implies that the representatives $\iota_{X,E}(w'_1), \iota_{X,E}(w'_2)$ of w'_1, w'_2 in $M_{X,E}$ are distinct.

Discuss 854 example

7 Automated Theorem Proving

TODO: expand this sketch

Describe various automated theorem provers deployed in the project, with some statistics on performance:

- Z3 prover
- Vampire
- More elementary substitutions
- egg
- duper

Any comparative study of semi-automated methods with fully automated ones? In principle, the semi-automated approach could be more automated using a script or "agent" to call various theorem provers. See this discussion.

Draw upon the discussion here on the various ways of integrating ATPs with Lean. This project primarily used the least integrated approach, "Option 3", as it was fastest and required no dependencies on the other contributors, but it also has drawbacks.

Note: when evaluating the performance of any particular automated implication tool, we should do a fresh run on the entire base of implications, rather than rely on whatever implications produced by the tool remain in the Lean codebase by the time of writing the paper. This is because (a) many of the previous runs focused only on those implications that were not already ruled out by earlier contributions, and (b) some pruning has been applied subsequent to the initial runs to improve compilation performance. Of course, these runs would not need to be added to the (presumably optimized) codebase at that point, but would be purely for the purpose of gathering performance statistics. More discussion here.

See this discussion on the value of using different ATPs and setting run time parameters etc. at different values.

Make a note of the possible alternate strategy to prove implications outlined here.

8 Implications for Finite Magmas

TODO: Expand this sketch

Introduce Austin pairs.

Recap discussion from `https://leanprover.zulipchat.com/#narrow/channel/458659-Equational/topic/Austin.20pairs`

9 Order 5 laws

TODO: report on laws on order 5

10 Higman–Neumann laws

TODO: report on Higman–Neumann laws

11 AI and Machine Learning Contributions

TODO: expand this sketch

- Claude assistance in coding front ends.
- ChatGPT to guess a complete rewriting system.
- Minor use of GitHub Copilot to autocomplete code in Lean and other languages.
- See this discussion.
- Directed Link Prediction (DLP) on the implication graph with multiple GNN autoencoders (see related Zulip topic).

ML experiments to learn the implication graph.

11.1 Graph ML: Directed link prediction on the implication graph

We experimented with various Graph Neural Network (GNN) autoencoders to predict missing edges, providing a way to estimate the truth value of unproven implications. To assess these models, we defined three test sets focusing on edge existence, directionality, and bidirectionality. The results give insight into how these models handle dense, directed graphs like ours. A more detailed report follows below.

11.1.1 Motivation

Directed Link Prediction [8] is a method enabling machine learning and deep learning models to predict missing edges in a directed graph. For our implication graph, this translates to predicting the truth values of unproven implications. This task serves as a necessary first step for advancing in the following directions:

1. **Reasoning over mathematical knowledge graphs:** Recent advancements allow language models to integrate information from multiple modalities. For example, [27, 26] share information between corresponding layers of Language Models (LMs) and Graph Neural Networks (GNNs), enabling simultaneous learning from text corpora and graph-structured data within the same expert domain. By leveraging both modalities, the language model can better *structure* its knowledge and respond to complex queries. This dual learning process combines masked language modeling for text with link prediction on the graph, highlighting the importance of link prediction for robust reasoning.
2. **Higher-order implication graphs:** Our implication graph currently represents only implications of the form $p \implies q$, not more complex ones like $(p \wedge r) \implies q$. Extending to such higher-order edges would likely involve connecting sets of nodes, thereby requiring hypergraph representations. For a systematic overview, see [10]. While specific hypergraph neural architectures exist [5], we believe it is still conceptually important to apply Directed Link Prediction to simpler implication graphs first, providing insights and guiding principles that can anticipate challenges in higher-order graph representation learning.

11.1.2 Data

We used this edge list, generated on October 20, 2024, with the following commands:

```
lake exe extract_implications outcomes > data/tmp/outcomes.json
scripts/generate_edgelist_csv.py
```

The structure of the implication graph is summarized below:

```
graph_summary = {
    "total_nodes": 4694,
    "total_directed_edges": 8178279,
    "edge_density_percentage": 37,  # Percentage of possible edges that exist
    "bidirectional_edges": 2475610,
    "bidirectional_percentage": 30  # Percentage of all edges that are bidirectional
}
```

Below is a summary of the edge types in the graph:

```
edge_counts = {
    "explicit_conjecture_false": 92,
    "explicit_proof_false": 582316,
    "explicit_proof_true": 10657,
    "implicit_conjecture_false": 142,
    "implicit_proof_false": 13272681,
    "implicit_proof_true": 8167622,
    "unknown": 126
}
```

Edges are labeled according to the following scheme:

```
edge_labels = {
    "implicit_proof_true": 1,
    "explicit_proof_false": 0,
    "implicit_proof_false": 0,
    "explicit_proof_true": 1,
    "explicit_conjecture_false": 0,
    "implicit_conjecture_false": 0,
    "unknown": 0
}
```

The `unknown` class contains a very small number of edges (126), so their impact on the training phase is expected to be negligible. Future approaches might address this class by excluding `unknown` edges from the training set.

11.1.3 Methods

Consider a directed graph $G = (V, E)$ where $E = \{(u, v) \mid u, v \in V\}$ is the edge set and $|V| = n$. We assume that each node is associated with a feature vector, resulting in an $X \in \mathbb{R}^{n \times f}$ feature matrix.

We define the existing edges $(a, b) \in E$ as *positives* and the non-existing edges $(c, d) \notin E$ as *negatives*.

Intuitively, performing Directed Link Prediction (DLP) on G involves randomly splitting E into three disjoint sets: E_{train}, E_{val}, and E_{test}, such that:

- E_{train} is the training set,
- E_{val} is the validation set,
- E_{test} is the test set,
- and $E = E_{\text{train}} \dot{\cup} E_{\text{val}} \dot{\cup} E_{\text{test}}$.

The model then learns from $G_{\text{train}} = (V, E_{\text{train}})$ to map the topological and feature-related information of two nodes u and v to a probability p_{uv} that $(u, v) \in E_{\text{test}}$.

However, this setup presents two key issues among others:

1. The model learns only from *positives*, so it cannot recognize *negatives.*
2. The model is evaluated only on *positives*, preventing us from measuring its ability to identify *negatives.*

To address these limitations, we adopted the setup proposed in [23]. Specifically, we redefined E_{train} to E^{p}_{train} (*positives*) and introduced:

$$E_{\text{train}} = E^{p}_{\text{train}} \dot{\cup} E^{n}_{\text{train}}$$

where E^{n}_{train} includes all possible *negatives* in $G_{\text{train}} = (V, E^{p}_{\text{train}})$. The model is now required to predict the non-existence of edges in E^{n}_{train}.

Similarly, if we redefine E_{test} as follows:

$$E_{\text{test}} = E^{p}_{\text{test}} \dot{\cup} E^{n}_{\text{test}}$$

where E^{n}_{test} is a *random* sample of *negatives*, the model's evaluation would fail to capture two crucial aspects:

1. The model's ability to distinguish (u, v) from (v, u) for all $(u, v) \in E^p_{\text{test}}$.
2. The model's ability to identify bi-implications.

These limitations arise from the random selection of negative edges in E^n_{test}. To address this, we define three distinct test sets: E^G_{test}, E^D_{test}, and E^B_{test}, to evaluate different facets of the model's performance:

- **General Test Set** (E^G_{test}): Here, $E_{\text{test}} = E^p_{\text{test}} \dot{\cup} E^n_{\text{test}}$, where E^n_{test} is a random sample of non-existent edges with the same cardinality as E^p_{test}. This set assesses the model's ability to detect the presence of edges, regardless of direction. A model that cannot distinguish edge direction may still perform well on this set, highlighting the need for the following two additional test sets.
- **Directional Test Set** (E^D_{test}): Defined as $E^{\text{up}}_{\text{test}} \dot{\cup} \tilde{E}^{\text{up}}_{\text{test}}$, where $E^{\text{up}}_{\text{test}}$ consists of unidirectional edges in E^p_{test}, and $\tilde{E}^{\text{up}}_{\text{test}}$ contains their reverses (negatives by construction). This set evaluates the model's ability to recognize edge direction, making it suitable for assessing direction-sensitive models.
- **Bidirectional Test Set** (E^B_{test}): Defined as $E^{\text{bp}}_{\text{test}} \dot{\cup} E^{\text{n}}_{\text{B}}$, where $E^{\text{bp}}_{\text{test}}$ contains one direction of all bidirectional edges in E^p_{test}, and $E^{\text{n}}_{\text{B}} \subset \tilde{E}$ includes a subset of their reverses. This set evaluates the model's ability to identify bi-implications, as each edge in E^B_{test} has a reverse that is positive, but only half are bidirectional in practice.

We tested the following models:

- **GAE** [8]
- **Gravity-GAE** [23]
- **Source/Target-GAE** [23]
- **DiGAE** [13]
- **MagNet** [28]

All these models are graph-based autoencoders with distinct encoder-decoder architectures. Notably, GAE is the only model structurally unable to differentiate edge directions. Each model outputs the probability that an ordered pair of nodes has a directed edge between them, with nodes represented using one-hot encodings as features.

We trained the models using Binary Cross Entropy as the loss function, balancing the contribution of positive and negative edges through re-weighting. On the *General* test set, we evaluated the following metrics:

- **AUC** (Area Under the ROC Curve): Measures the probability that the model ranks a random positive edge higher than a random negative edge. Higher values indicate better discrimination between positive and negative edges.
- **AUPRC** (Area Under Precision-Recall Curve): Assesses model performance, particularly in cases of class imbalance. AUPRC is more robust to imbalanced data than AUC.
- **Hits@K**: Evaluates the fraction of times a positive edge is ranked within the top K positions among personalized negative samples [15]. Briefly, given a positive edge, its M personalized negative samples are M negative edges with the same head but different tails. We calculate Hits@K for $K = 1, 3, 10$ to assess ranking quality, and set $M = 100$. Therefore, Hits@K = 0.1 means that on average, the model ranks a positive edge within the highest-ranked K personalized negatives 10% of the time.
- **MRR** (Mean Reciprocal Rank): Computes the average reciprocal rank of positive edges among their personalized negative samples [15] (the same as those used for Hits@K) providing an overall measure of ranking accuracy. For instance, $MRR = 0.1$ means that on average, the model ranks a positive edge as 10^{th} among M personalized negative samples.

Each metric ranges from 0 to 1, with higher values reflecting improved performance. Based on prior work, we expect AUC and AUPRC scores to approach 1, while MRR and Hits@K often yield values around 0.15 for similar undirected tasks [15]. *Directional* and *Bidirectional* performances were evaluated using only AUC and AUPRC, since Hits@K and MRR are hardly definable in those scenarios. The number of training epochs was optimized through Early Stopping on the *General* validation set performance (given by the sum of AUC and AUPRC).

11.1.4 Results

The results below represent average performance over six random splits of E_{train}, E_{val}, and E_{test} while keeping the model's seed fixed for fair comparison. The *training / validation / test* split proportions are:

- 85/5/10 for unidirectional edges,
- 65/15/30 for bidirectional edges.

Model	**G_ROC_AUC**	**G_AUPRC**	**G_Hits@1**	**G_Hits@3**
`gae`	$0.8484 \pm 9 \times 10^{-4}$	$0.8558 \pm 6 \times 10^{-4}$	$6 \times 10^{-5} \pm 4 \times 10^{-5}$	$6 \times 10^{-5} \pm 4 \times 10$
`gravity_gae`	$0.9806 \pm 3 \times 10^{-4}$	$0.9753 \pm 4 \times 10^{-4}$	$0.069 \pm 6 \times 10^{-3}$	$0.101 \pm 5 \times 10^{-}$
`sourcetarget_gae`	$0.99976 \pm 1 \times 10^{-5}$	$0.999736 \pm 8 \times 10^{-6}$	$0.077 \pm 4 \times 10^{-3}$	$0.147 \pm 7 \times 10^{-}$
`mlp_gae`	$0.99315 \pm 1 \times 10^{-5}$	$0.99409 \pm 1 \times 10^{-5}$	$0.181 \pm 7 \times 10^{-3}$	$0.299 \pm 7 \times 10^{-}$
`digae`	$0.9978 \pm 3 \times 10^{-4}$	$0.998 \pm 3 \times 10^{-4}$	$0.035 \pm 6 \times 10^{-3}$	$0.068 \pm 1 \times 10^{-}$
`magnet`	$0.989 \pm 1 \times 10^{-4}$	$0.99076 \pm 3 \times 10^{-5}$	$0.151 \pm 1 \times 10^{-2}$	$0.26 \pm 2 \times 10^{-2}$

Table 1: Results for various graph autoencoder models.

11.1.5 Discussion

We observe consistently high *General* AUC and AUPRC scores, close to 1. These high values are expected, as similar neural architectures have demonstrated strong performance in graphs of comparable size [8]. The high ratio of existing to non-existing edges in the implication graph (approximately 37%) likely contributes to the near-perfect *General* AUC and AUPRC scores. For context, benchmark datasets such as Cora and Citeseer (e.g., directed and undirected) contain fewer than 1% of all possible edges.

Interestingly, the GAE model, though structurally unable to distinguish edge direction, performs well on the *General* task (if we consider AUC and AUPRC only). This experimentally confirms the need for including *Directional* and *Bidirectional* test sets, which allow comprehensive evaluation across all facets of Directed Link Prediction (DLP).

All other models demonstrate high AUC and AUPRC scores across the *General*, *Directional*, and *Bidirectional* test sets, indicating strong predictive capabilities even when accounting for directionality and bidirectionality.

Notably, the `mlp_gae` and `magnet` models also achieve competitive scores in MRR and Hits@K metrics.

11.1.6 Conclusions

We evaluated the performance of six different graph autoencoders on a Directed Link Prediction (DLP) task. By adopting a multi-task evaluation framework, we assessed the models comprehensively across various aspects of DLP. All models demonstrated strong performance on AUC and AUPRC metrics, with some also achieving high scores on MRR and Hits@K.

Node features were represented using one-hot encodings, meaning that the models received no explicit information about the equations represented by the nodes. Instead, they relied entirely on the topological structure encoded during training. This approach aligns with previous research suggesting that one-hot encodings can promote asymmetric embeddings [23]. However, future experiments could explore alternative representations, such as encoding the equations with text-based embeddings like Word2Vec, to potentially enhance the models' understanding of the nodes' semantic content.

In summary, our findings highlight the adaptability and robustness of graph autoencoders for DLP tasks in dense, directed graphs. This approach not only demonstrates robustness in predicting directed links but also suggests promising potential for future improvements through enhanced feature representations, thereby advancing the capabilities of link prediction in complex mathematical knowledge graphs.

12 User Interface

Describe visualizations and explorer tools: Equation Explorer, Finite Magma Explorer, Graphiti, ...

13 Statistics and Experiments

TODO: Expand this sketch

Data analysis of the implication graph:

- Mention the long chain $2 \Rightarrow 5 \Rightarrow 2499 \Rightarrow 2415 \Rightarrow 238 \Rightarrow 2716 \Rightarrow 28 \Rightarrow 2973 \Rightarrow 270 \Rightarrow 3 \Rightarrow 3715 \Rightarrow 375 \Rightarrow 359 \Rightarrow 4065 \Rightarrow 1$ (discussed here).
- What are the "most difficult" implications?
- Is there a way to generate a standard test set of implication problems of various difficulty levels? Can one then use this to benchmark various automated and semi-automated methods? Challenge: how does one automatically assign a difficulty level to a given (anti-)implication?

See this for a preliminary data analysis of the impact of equation size and the number of variables.

Analyze the implication graph and discuss test sets of implication problems for benchmarking theorem provers. Challenge: How can one automatically assign a difficulty level to an implication?

14 Data Management

TODO: expand this sketch

Describe how data was handled during the project and how it will be managed going forward.

15 Reflections

TODO: expand this sketch

Testimonies from individual participants (but perhaps this is more suited for a retrospective paper). Utilize the thoughts and reflections thread.

Automation often overtook the rate of human progress, for instance in developing metatheorems. Does this create an opportunity cost? Raised as a possible discussion point here by Zoltan Kocsis.

16 Conclusions and Future Directions

TODO: Expand this sketch

Insights learned, future speculation. Utilize the discussion on future directions. Some ideas from that list:

- A database of triple implications (EquationX and EquationY imply EquationZ) - see also this discussion.
- Are there any equational laws that have no non-trivial finite models, but have surjunctive models?
- Can we produce interesting examples of irreducible implications EquationX -¿ EquationY (no intermediate EquationZ can interpose)
- Degree of satisfiability results, e.g., if a central groupoid obeys the natural central groupoid axiom 99% of the time, is it a natural central groupoid?
- Kisielewicz's question: does 5093 have any infinite models?
- "Insight mining" the large corpus of automated proofs that have been generated.
- How machine-learnable is the implication graph? (See AI/ML section)

Acknowledgments

Thanks to Claudio Moroni for the exploration of directed link prediction on the implication graph using GNN autoencoders described in Section 11.1.

Acknowledgments to the broader Lean Zulip community and smaller contributors not listed as authors.

A Numbering system

In this section we record the numbering conventions we use for equational laws.

For this formal definition we use the natural numbers $0, 1, 2, \dots$ to represent and order indeterminate variables; however, in the main text, we use the symbols $x, y, z, w, u, v, r, s, t$ instead (and do not consider any laws with more than eight variables).

We requiring extend the ordering on indeterminate variables to a well-ordering on words w in the free magma generated by these variables by declaring $w > w'$ if one of the following holds:

- w has a larger order than w'.
- $w = w_1 \diamond w_2$ and $w' = w'_1 \diamond w'_2$ have the same order $n \geq 1$ with $w_1 > w'_1$.
- $w = w_1 \diamond w_2$ and $w' = w'_1 \diamond w'_2$ have the same order $n \geq 1$ with $w_1 = w'_1$ and $w_2 > w'_2$.

Thus for instance

$$0 < 1 < 0 \diamond 0 < 0 \diamond 1 < 1 \diamond 0$$

and

$$1 \diamond 1 < 0 \diamond (0 \diamond 0) < (0 \diamond 0) \diamond 0.$$

We similarly place a well-ordering on equational laws $w_1 \simeq w_2$ by declaring $w_1 \simeq w_2 > w'_1 \simeq w'_2$ if one of the following holds: as follows:

- $w_1 \simeq w_2$ has a longer order than $w'_1 \simeq w'_2$.
- If $w_1 \simeq w_2$ has the same order as $w'_1 \simeq w'_2$, and $w_1 > w'_1$.
- If $w_1 \simeq w_2$ has the same order as $w'_1 \simeq w'_2$, $w_1 = w'_1$, and $w_2 > w'_2$.

Two equational laws are equivalent if one can be obtained from another by some combination of relabeling the variables and applying the symmetric law $w_1 \simeq w_2 \iff w_2 \simeq w_1$. For instance, $(0 \diamond 1) \diamond 2 \simeq 1$ is equivalent to $0 \simeq (1 \diamond 0) \diamond 2$. We then replace every equational law with their minimal element in their equivalence class, which can be viewed as the *normal form* for that law; for instance, the normal form of $(0 \diamond 1) \diamond 2 \simeq 1$ would be $0 \simeq (1 \diamond 0) \diamond 2$. Finally, we eliminate any law of the form $w \simeq w$ other than $0 \simeq 0$. We then number the remaining equations $E1, E2, \dots$. For instance, $E1$ is the trivial law $0 \simeq 0$, $E2$ is the constant law $0 \simeq 1$, $E3$ is the idempotent law $0 \simeq 0 \diamond 0$, and so forth. Lists and code for generating these equations, or the equation number attached to a given equation, can be found at the ETP repository.

The number of equations in this list of order $n = 0, 1, 2, \dots$ is given by

$$2, 5, 39, 364, 4284, 57882, 888365, \dots$$

(`https://oeis.org/A376640`). The number can be computed to be

$$C_{n+1} B_{n+2}/2$$

if n is odd, 2 if $n = 0$, and

$$(C_{n+1} B_{n+2} + C_{n/2}(2D_{n+2} - B_{n+2}))/2 - C_{n/2} B_{n/2+1}$$

if $n > 2$ is even, where C_n, B_n are the Catalan and Bell numbers, and D_n is the number of partitions of $[n]$ up to reflection, which for $n = 0, 1, 2, \dots$ is

$$1, 1, 2, 4, 11, 32, 117, \dots$$

(`https://oeis.org/A103293`). A proof of this claim can be found in the ETP blueprint. In particular, there are 4694 equations of order at most 4.

Below we record some specific equations appearing in this paper, using the alphabet x, y, z, w, u, v

in place of $0, 1, 2, 3, 4, 5, \ldots$ for readability.

$$
\begin{array}{rlr}
x = x & \text{(Trivial law)} & \text{(E1)} \\
x = y & \text{(Singleton law)} & \text{(E2)} \\
x = x \diamond x & \text{(Idempotent law)} & \text{(E3)} \\
x = x \diamond y & \text{(Left-absorptive law)} & \text{(E4)} \\
x = y \diamond x & \text{(Right-absorptive law)} & \text{(E5)} \\
x = (x \diamond x) \diamond x & & \text{(E23)} \\
x \diamond x = y \diamond z & & \text{(E41)} \\
x \diamond y = y \diamond x & \text{(Commutative law)} & \text{(E43)} \\
x \diamond y = z \diamond w & \text{(Constant law)} & \text{(E46)} \\
x = (y \diamond x) \diamond (x \diamond z) & \text{(Central groupoid law)} & \text{(E168)} \\
x \diamond y = x \diamond (y \diamond z) & & \text{(E327)} \\
x \diamond y = (z \diamond x) \diamond y & & \text{(E395)} \\
x = y \diamond ((z \diamond (x \diamond (y \diamond z)))) & \text{(Tarski's axiom)} & \text{(E543)} \\
x = x \diamond ((y \diamond z) \diamond (x \diamond z)) & & \text{(E854)} \\
x = y \diamond ((y \diamond (x \diamond z)) \diamond z) & & \text{(E1117)} \\
x = y \diamond (((x \diamond y) \diamond x) \diamond y) & & \text{(E1286)} \\
x = (y \diamond z) \diamond (y \diamond (x \diamond z)) & \text{(Exp. 2 abelian groups)} & \text{(E1571)} \\
x = (y \diamond x) \diamond ((x \diamond z) \diamond z) & & \text{(E1689)} \\
x = (x \diamond ((x \diamond x) \diamond x)) \diamond x & & \text{(E2441)} \\
(x \diamond y) \diamond z = x \diamond (y \diamond z) & \text{(Associative law)} & \text{(E4512)} \\
x \diamond (y \diamond z) = (y \diamond z) \diamond x & & \text{(E4531)} \\
x = (y \diamond ((x \diamond y) \diamond y)) \diamond (x \diamond (z \diamond y)) & \text{(Sheffer stroke)} & \text{(E345169)} \\
x = y \diamond ((((y \diamond y) \diamond x) \diamond z) & \text{(Division in groups)} & \text{(E42323216)} \\
\quad \diamond (((y \diamond y) \diamond y) \diamond z)) & &
\end{array}
$$

B Proofs of Theoretical Results

Provide the interesting proofs mentioned in the results section, while routine proofs can refer to the blueprint or Lean.

C Author Contributions

In this section we list the authors of this paper, grant support, and their contributions to this project, using the following standard CRediT categories:

1. Conceptualization: Ideas; formulation or evolution of overarching research goals and aims.
2. Data Curation: Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later reuse.
3. Formal Analysis: Application of statistical, mathematical, computational, or other formal techniques to analyze or synthesize study data.
4. Funding Acquisition: Acquisition of the financial support for the project leading to this publication.
5. Investigation: Conducting a research and investigation process, specifically performing the experiments, or data/evidence collection.
6. Methodology: Development or design of methodology; creation of models.
7. Project Administration: Management and coordination responsibility for the research activity planning and execution.
8. Resources: Provision of study materials, reagents, materials, patients, laboratory samples, animals, instrumentation, computing resources, or other analysis tools.
9. Software: Programming, software development; designing computer programs; implementation of the computer code and supporting algorithms; testing of existing code components.
10. Supervision: Oversight and leadership responsibility for the research activity planning and execution, including mentorship external to the core team.
11. Validation: Verification, whether as a part of the activity or separate, of the overall replication/reproducibility of results/experiments and other research outputs.
12. Visualization: Preparation, creation and/or presentation of the published work, specifically visualization/data presentation.
13. Writing - Original Draft Preparation: Creation and/or presentation of the published work, specifically writing the initial draft (including substantive translation).
14. Writing – Review and Editing: Preparation, creation and/or presentation of the published work by those from the original research group, specifically critical review, commentary or revision – including pre- or post-publication stages.

- ...
- Pietro Monticone, Department of Mathematics, University of Trento, pietro.monticone@studenti.unit
Data Curation, Formal Analysis, Project Administration, Resources, Software, Validation, Writing - Original Draft Preparation, Writing - Review and Editing.

- Terence Tao, Department of Mathematics, UCLA, tao@math.ucla.edu: Conceptualization, Data Annotation, Investigation, Methodology, Project Administration, Writing - Original Draft Preparation, Writing - Review and Editing. Supported by NSF grant DMS-2347850.

- ...

References

[1] A. K. Austin. A note on models of identities. *Proc. Amer. Math. Soc.*, 16:522–523, 1965.

[2] Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, Cambridge, 1998.

[3] George M Bergman. The diamond lemma for ring theory. *Advances in Mathematics*, 29(2):178–218, 1978.

[4] Trevor Evans. Products of points—some simple algebras and their identities. *Amer. Math. Monthly*, 74:362–372, 1967.

[5] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3558–3565, July 2019.

[6] Timothy Gowers and Michael Nielsen. Massively collaborative mathematics. *Nature*, 461(7266):879–881, October 2009.

[7] Graham Higman and B. H. Neumann. Groups as groupoids with one law. *Publ. Math. Debrecen*, 2:215–221, 1952.

[8] Thomas N. Kipf and Max Welling. Variational graph auto-encoders. 2016.

[9] A. Kisielewicz. Austin identities. *Algebra Universalis*, 38(3):324–328, 1997.

[10] M. Kivela, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, July 2014.

[11] Donald E. Knuth. Notes on central groupoids. *J. Combinatorial Theory*, 8:376–390, 1970.

[12] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In *Computational Problems in Abstract Algebra (Proc. Conf., Oxford, 1967)*, pages 263–297. Pergamon, Oxford-New York-Toronto, Ont., 1970.

[13] Georgios Kollias, Vasileios Kalantzis, Tsuyoshi Idé, Aurélie Lozano, and Naoki Abe. Directed graph auto-encoders. 2022.

[14] André Kündgen, Gregor Leander, and Carsten Thomassen. Switchings, extensions, and reductions in central digraphs. *J. Combin. Theory Ser. A*, 118(7):2025–2034, 2011.

[15] Juanhui Li, Harry Shomer, Haitao Mao, Shenglai Zeng, Yao Ma, Neil Shah, Jiliang Tang, and Dawei Yin. Evaluating graph neural networks for link prediction: Current pitfalls and new benchmarking. 2023.

[16] William McCune. Solution of the Robbins problem. *J. Automat. Reason.*, 19(3):263–276, 1997.

[17] William McCune. Single axioms: with and without computers. In *Computer mathematics (Chiang Mai, 2000)*, volume 8 of *Lecture Notes Ser. Comput.*, pages 83–89. World Sci. Publ., River Edge, NJ, 2000.

[18] William McCune, Robert Veroff, Branden Fitelson, Kenneth Harris, Andrew Feist, and Larry Wos. Short single axioms for Boolean algebra. *J. Automat. Reason.*, 29(1):1–16, 2002.

[19] Ralph McKenzie. On spectra, and the negative solution of the decision problem for identities having a finite nontrivial model. *J. Symbolic Logic*, 40:186–196, 1975.

[20] N. S. Mendelsohn and R. Padmanabhan. Minimal identities for Boolean groups. *J. Algebra*, 34:451–457, 1975.

[21] C. A. Meredith and A. N. Prior. Equational logic. *Notre Dame J. Formal Logic*, 9:212–226, 1968.

[22] J. D. Phillips and Petr Vojt ˇ echovský. The varieties of loops of Bol-Moufang type. *Algebra Universalis*, 54(3):259–271, 2005.

[23] Guillaume Salha, Stratis Limnios, Romain Hennequin, Viet Anh Tran, and Michalis Vazirgiannis. Gravity-inspired graph autoencoders for directed link prediction. 2019.

[24] Alfred Tarski. Ein beitrag zur axiomatik der abelschen gruppen. *Fundamenta Mathematicae*, 30(1):253–256, 1938.

[25] Stephen Wolfram. The physicalization of metamathematics and its implications for the foundations of mathematics, Mar 2022.

[26] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy Liang, and Jure Leskovec. Deep bidirectional language-knowledge graph pretraining. 2022.

[27] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering. 2022.

[28] Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew Hirn. Magnet: A neural network for directed graphs. 2021.